REMARKS

This RESPONSE UNDER 37 CFR 1.116 is filed in reply to the outstanding Office Action of November 20, 2003, and is believed to be fully responsive thereto and to place this case in condition for allowance for reasons set forth below in greater detail.

Reconsideration is respectfully requested of the rejection of claims 1-4 as being allegedly obvious over Platko et al in view of Kozlowski et al, particularly in view of the following comments on the distinctions of each of the present invention, Platko and Kozlowski.

The Present Invention

The problem that the present invention seeks to solve is to allow a master high performance processor to access a high latency memory without stalling the processor. The solution is to use a slave processor to access the memory, freeing the master processor to continue other processing operations and tasks while the slave processor accesses the memory. The slave processor then posts the results of the reads to the master processor later.

Platko et al

The cited prior art Platko interfaces a slave processor (e.g. encryption engine) to an ASIC (master processor) at a lower cost, by reusing the memory bus and employing some extra signals. Reusing the memory bus reduces the number of pins consumed on the ASIC.

However, the Platko reference does not allow the master processor to continue other processing operations and tasks while the slave processor is reading the memory; indeed, Platko requires that the master processor in one scenario actually drive the reads that generate data onto the memory bus, and then further drives the writes that put the data into the slave processor, and in another scenario use reads and writes to transfer the data from slave processor to memory.

Though the same terms are used in the present invention and Platko, the "master processor" in the present invention is very different from the "master processor" in the prior art

Platko reference which is actually an ASIC with custom logic. In Platko, only a portion of this custom logic need be involved in data transfers, and it is therefore relatively inexpensive to stall in marshalling these data transfers. In contrast thereto, in the present invention the "master processor" is a high performance processor, which is entirely consumed while waiting for the data transfer, and is therefore expensive.

So, a key difference is that the master processor of the present invention is able to continue processing operations while the read is being performed by the slave processor.

Those features of the present invention are mentioned in the specification at page 8, lines 12-13, "In the meantime the processor is free to perform other tasks," and at page 11, lines 15-21, "Advantageously, the master processor 10 does not then have to wait or otherwise stall until the requested data is returned from the memory 50. This is because the master processor 10 has effectively delegated execution of the read request to the slave processor 20. The slave processor 30 absorbs any waiting, stalling or other latency associated with the memory access."

In the cited prior art reference Platko, the master processor is synchronously sequencing the operation of the slave processor. Whereas, in the present invention, the master processor is sending an asynchronous request to the slave processor and the slave processor is coming back with an asynchronous reply. In summary, the present invention avoids stalling the master processor while the asynchronous memory read is in progress.

Platko concerns, col. 1, lines 27-40, "a network interface card (NIC) of the type used in host systems such as personal computers and workstations. NICs are generally plug-in circuit cards having an interface to an I/O bus used in the host system, along with an interface to a physical network medium. In a NIC, it is common to employ random access memory (RAM) as temporary buffer storage for packets that have been received from the network or that are to be transmitted on the network. Along with the buffer RAM, the NIC contains a significant amount of complex logic for implementing the respective interfaces to the host I/O bus and the

network, and to move data along respective datapaths between the I/O bus and the buffer RAM, and between the network and the buffer RAM."

In Platko, col. 2, lines 12-16, "an optional co-processor is supported without requiring a separate interface on a master processor. High system performance is achieved, while device cost and complexity are reduced by keeping pin counts relatively low.

In Platko, col. 2, lines 30-49, "The master processor effects data transfers directly between the memory and the slave processor over the memory data bus...Thus, data flows directly between the memory and the slave processor without passing through the master processor. The only additional pins required by the master processor are the pins for the control signals to the slave processor."

Kozlowski et al

Kozlowski et al is in a totally different processing field from Platko and concerns, col. 1, lines 14-29, processing of "image checks, deposit slips and other types of bank documents in which...document processing systems commonly contain multiple microprocessor elements that are responsible for different tasks within the machine. In addition to their individual processing responsibilities, these processor elements are required to communicate with each other. Latency occurs when a processor foregoes its own processing functions because it is waiting for a response from another processor. The lost time created by latency therefore significantly compromises the performance of multiple processor systems and has generally resulted in increased complexity in an effort to address the problem."

Kozlowski addresses this problem, col. 2, lines 31-59, by using, "a master processor 10 and a slave processor 20. A multi-channel interface 30 is disposed between the slave processor 20 and the master processor 10. The multi-channel interface 30 has a low latency channel 32 for transferring low-latency information, and a high-throughput channel 31 for transferring high-throughput information. The use of multiple channels thus allows simultaneous transfers of different data types...Thus, if the data to be transferred comprises high-throughput information, the transfer will take place across the high-throughput channel 31.

Alternatively, if the data to be transferred comprises low-latency information, the transfer will take place across the low-latency channel 32."

Accordingly, Platko and Kozlowski are implemented in completely different processing fields and systems and have completely different objectives, which basically makes the systems noncombinable.

Platko concerns a network interface card as used in a PC, does not want a separate interface, and wants to reduce the number of pins consumed on an ASIC processor.

Kozlowski, on the other hand, uses a multi-channel interface having a low latency channel 32, a high throughput channel 31, and additional protocol flag interface channels 33.

In summary, the Platko system and the Kozlowski system are in totally different fields, address totally different problems, have totally different objectives and implement totally different processing systems, such that one skilled in the art would have absolutely no incentive or reason to attempt to combine the systems, as is attempted in the Final Rejection in a very apparent and blatant attempt to combine the references through hindsight.

In fact, neither Platko nor Kozlowski is even close to addressing the technology of the present invention.

This application is now believed to be in condition for allowance, a Notice of Allowance is respectfully requested. If the Examiner believes a telephone conference might

expedite prosecution of this case, it is respectfully requested that he call applicant's attorney at (516) 742-4343.

Respectfully submitted,

William C. Roch
Registration No. 24,972

Scully, Scott, Murphy & Presser
400 Garden City Plaza
Garden City, New York 11530
(516) 742-4343

WCR/jf